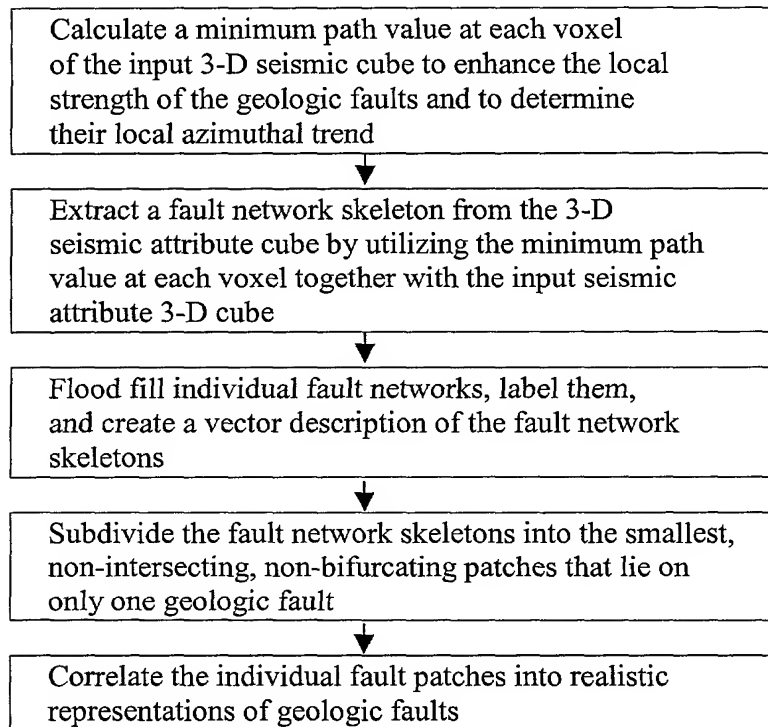


FIG. 1



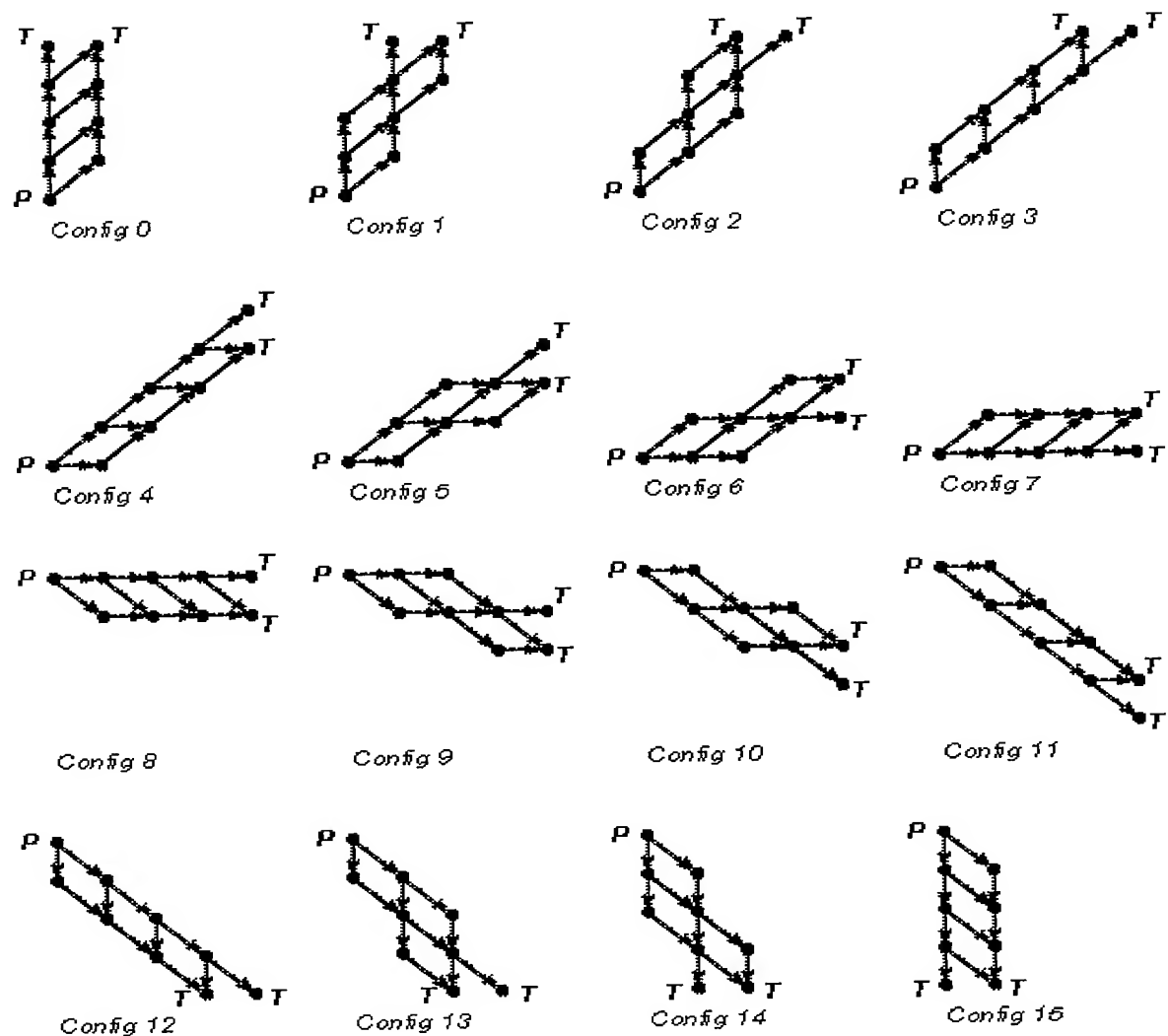


FIG. 2

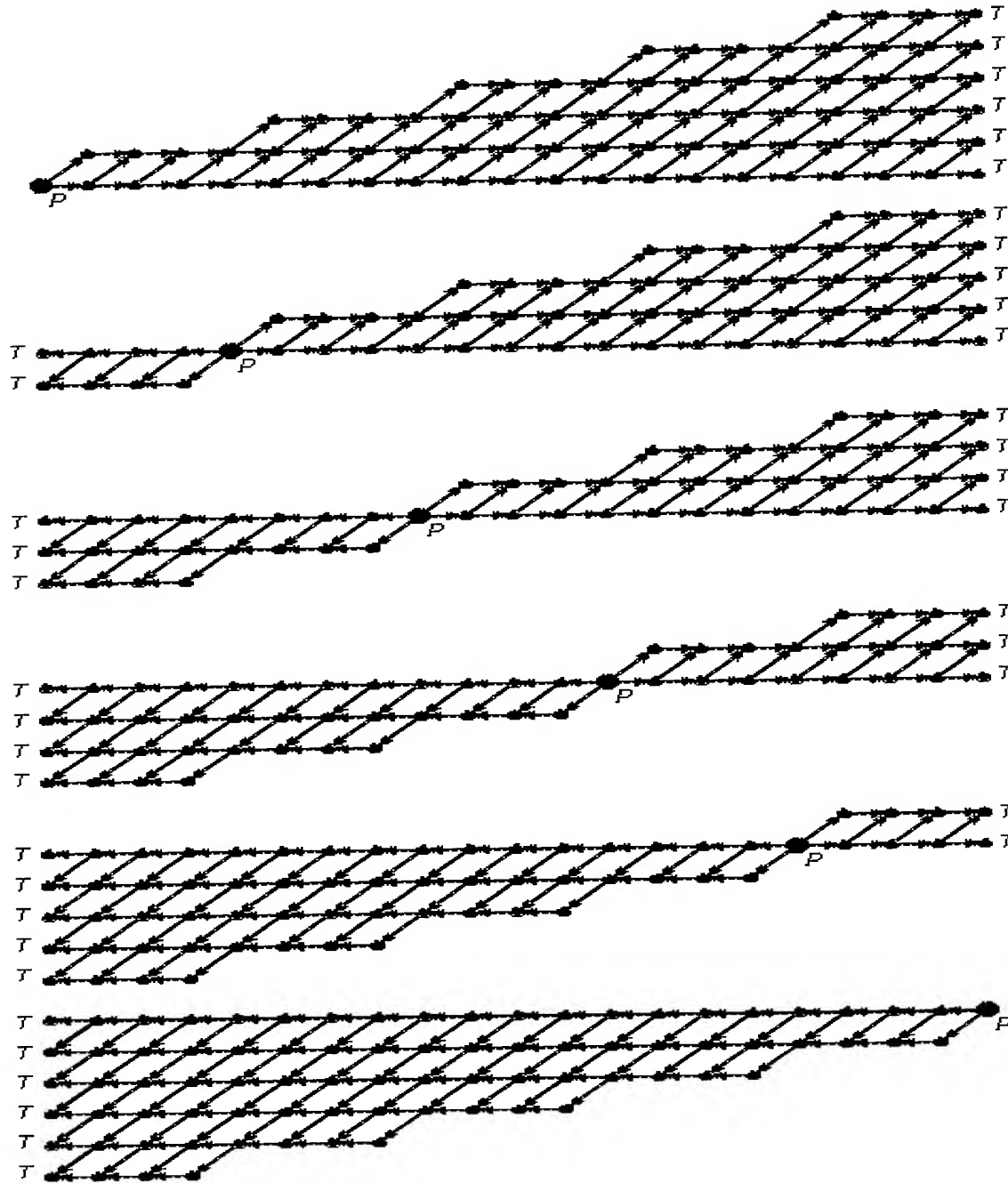


FIG. 3

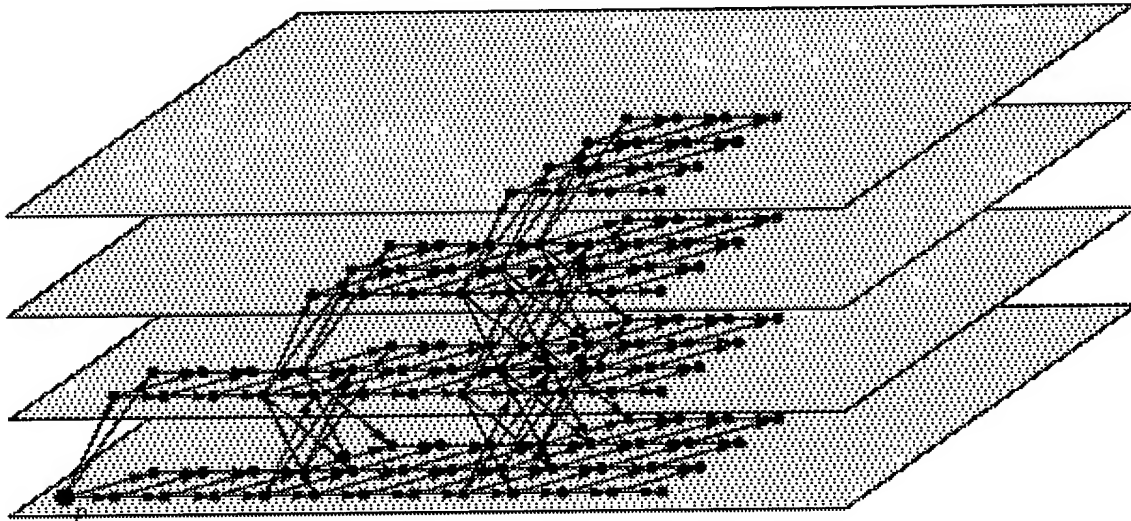


FIG. 4

FIG. 5

The algorithm proceeds one orientation at a time, and therefore goes through 16 iterations over the entire cube. After each iteration, the intermediate result is written to disk to achieve a memory efficient operation.

For a given orientation, the algorithm starts out by reading the $N+1$ first slices, indexed from 0 (first horizontal slice of the cube) to N . A data structure with $2*N+1$ entries, pointing to intermediate results for each slice, is then progressively filled. Initially only $N+1$ entries are represented in this structure.

After results for the current slice have been computed, this result is written to disk. If previous orientation results have already been computed and written to disk, the pixel-wise minimum of the current result for the current slice with the intermediate result computed for this slice thus far is taken.

A rotation on the aforementioned data structure so that the "middle" of this structure, that is, entry N , now points to intermediate results computed for slice number 1 of the cube is performed. An additional slice of the cube, that is, slice $N+1$ is read and incorporated into the structure. At this point, $N+2$ slices in the data structure has been computed.

After this shifting has taken place in the structure, the missing intermediate results needed for each entry, and the final results for slice number 1 for the current orientation are computed.

As before, results for slice 1 are written to disk and combined with previously stored intermediate results. A "rotation" is performed on the entries in the data structure, the next slice of the cube is read and incorporated to this structure.

When "steady state" has been reached, that is, when $2*N+1$ entries are in the data structure, each time the process proceeds down one slice and a rotation in the data structure entries is performed, a new slice is added at entry of index $2N$, and a slice and its associated intermediate results, that were computed for the first entry (index 0) in the structure, are also removed.

When the bottom of the cube is reached, no new slices can be added to the data structure after each rotation of its entries. The number of entries therefore decreases until there are only $N+1$ left in the structure. At this point, the entry of index N for the structure points to intermediate minimal path results corresponding to the last slice of the cube.

When results for this last slice have been computed and written to disk, this process repeats for the next orientation, until all 16 orientations have been considered.

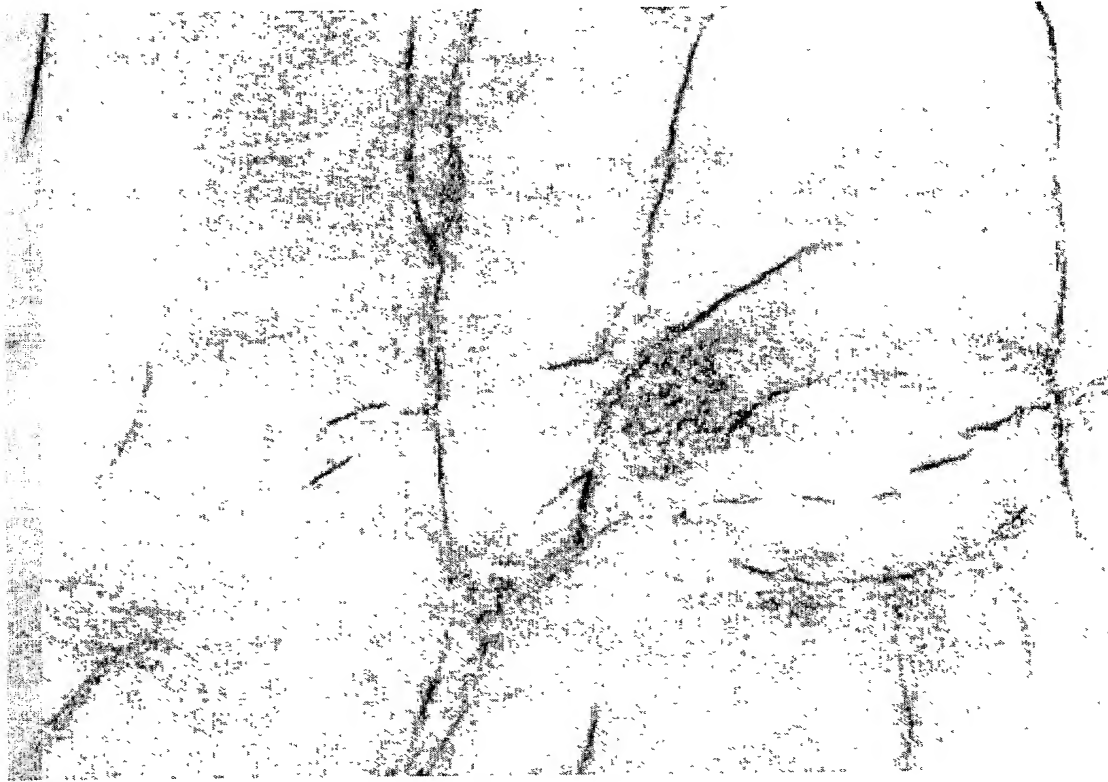


FIG. 6

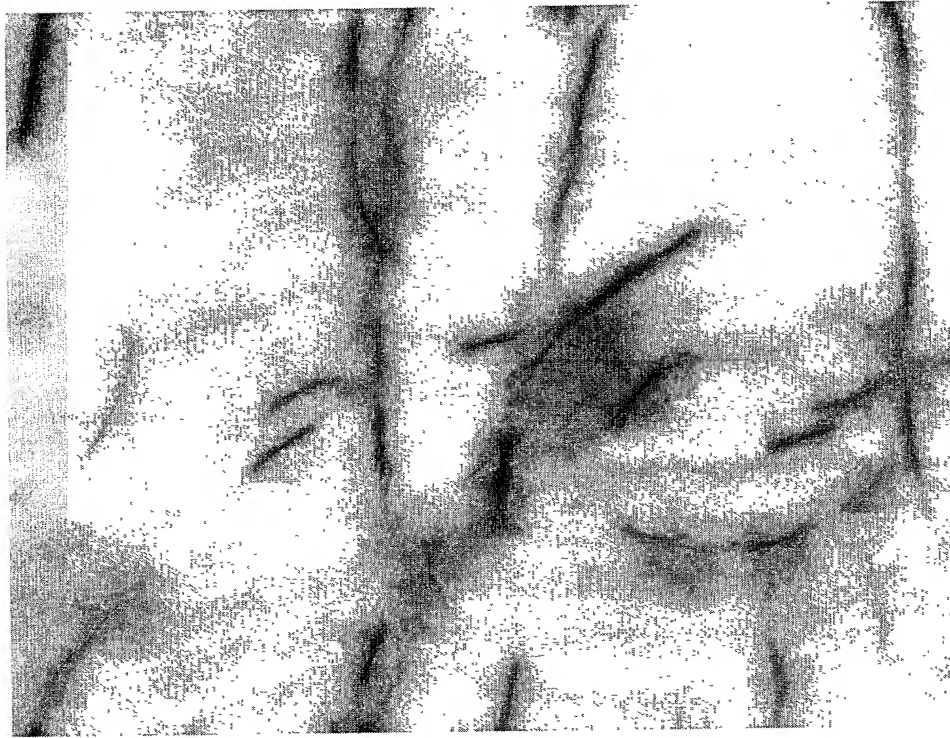


FIG. 7



FIG. 8

FIG. 9

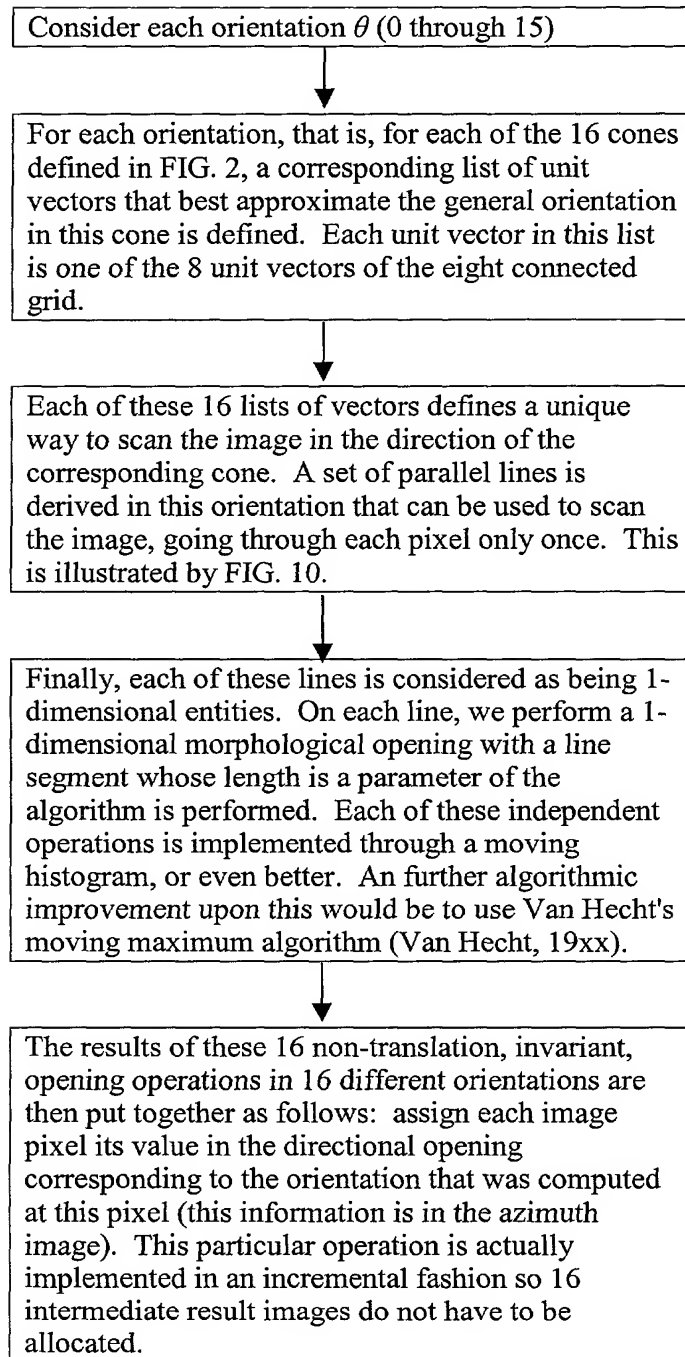


FIG. 9 is a flowchart

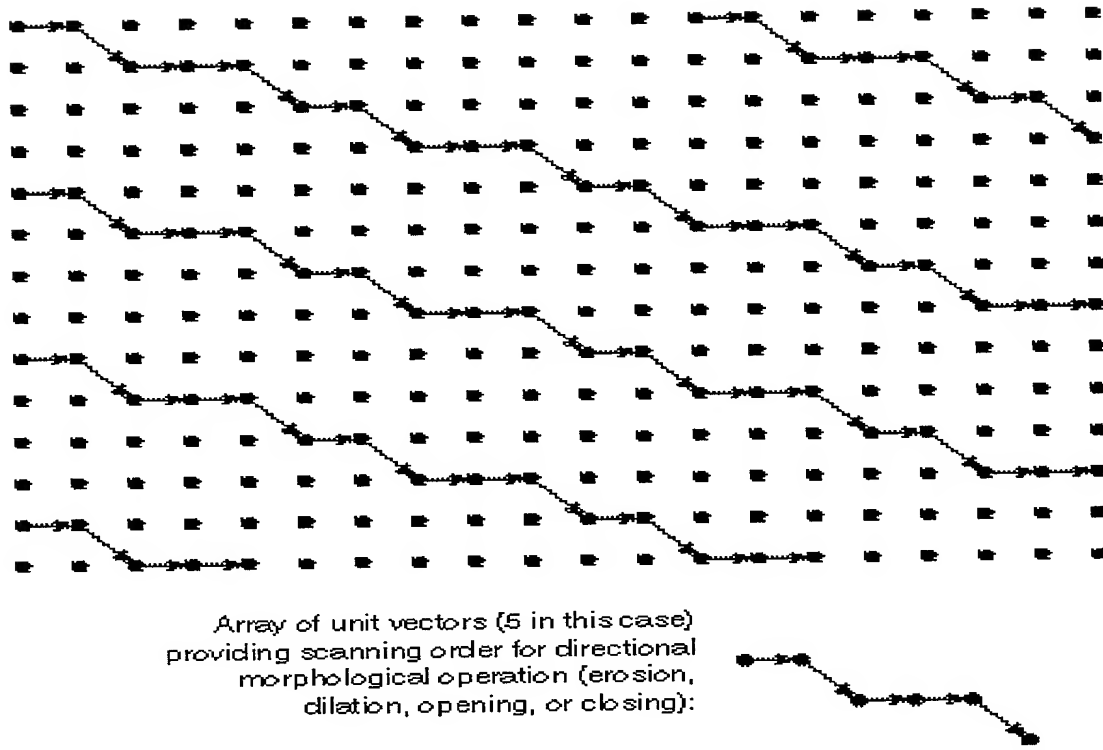


FIG. 10



FIG. 11



FIG. 12



FIG. 13

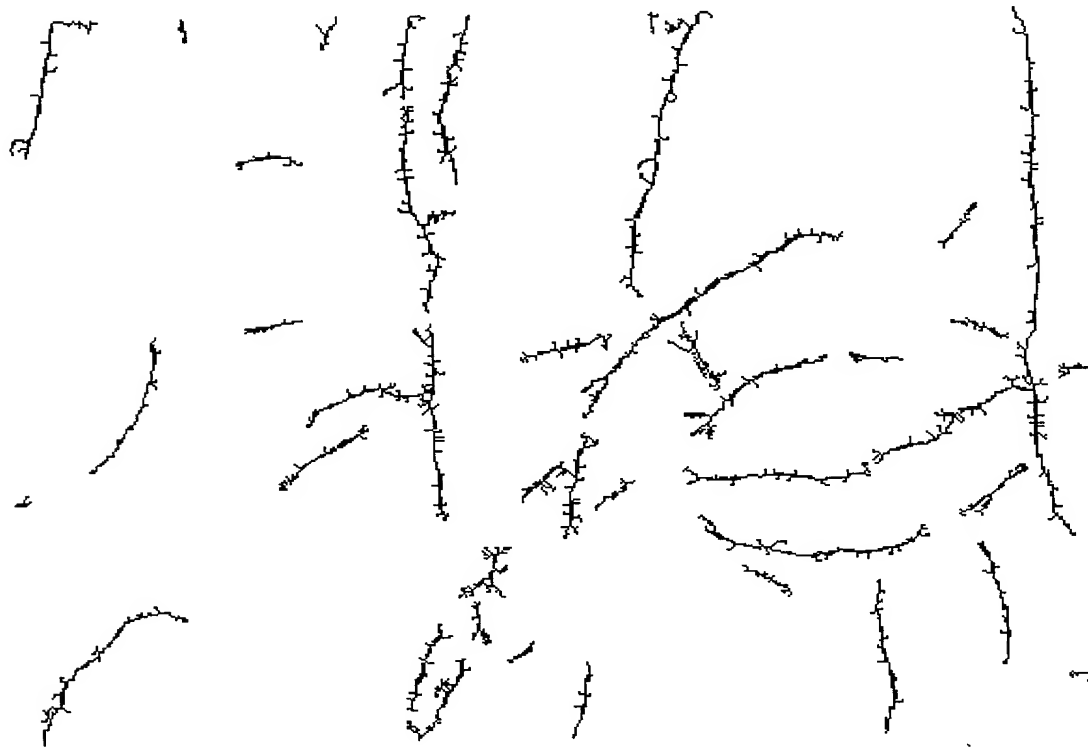


FIG. 14



FIG. 15



FIG. 16



FIG. 17

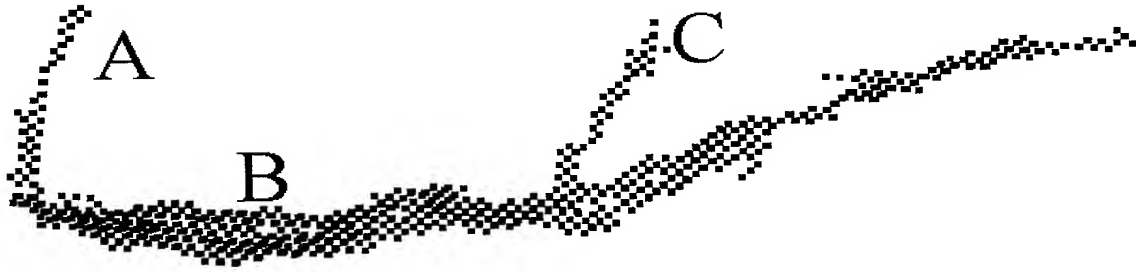


FIG. 18

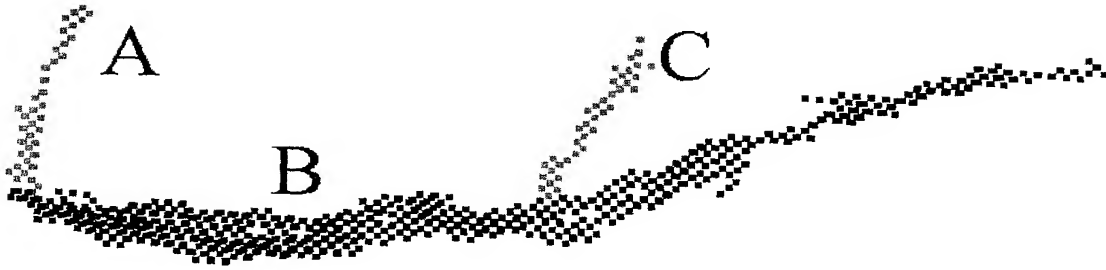


FIG. 19

FIG. 19

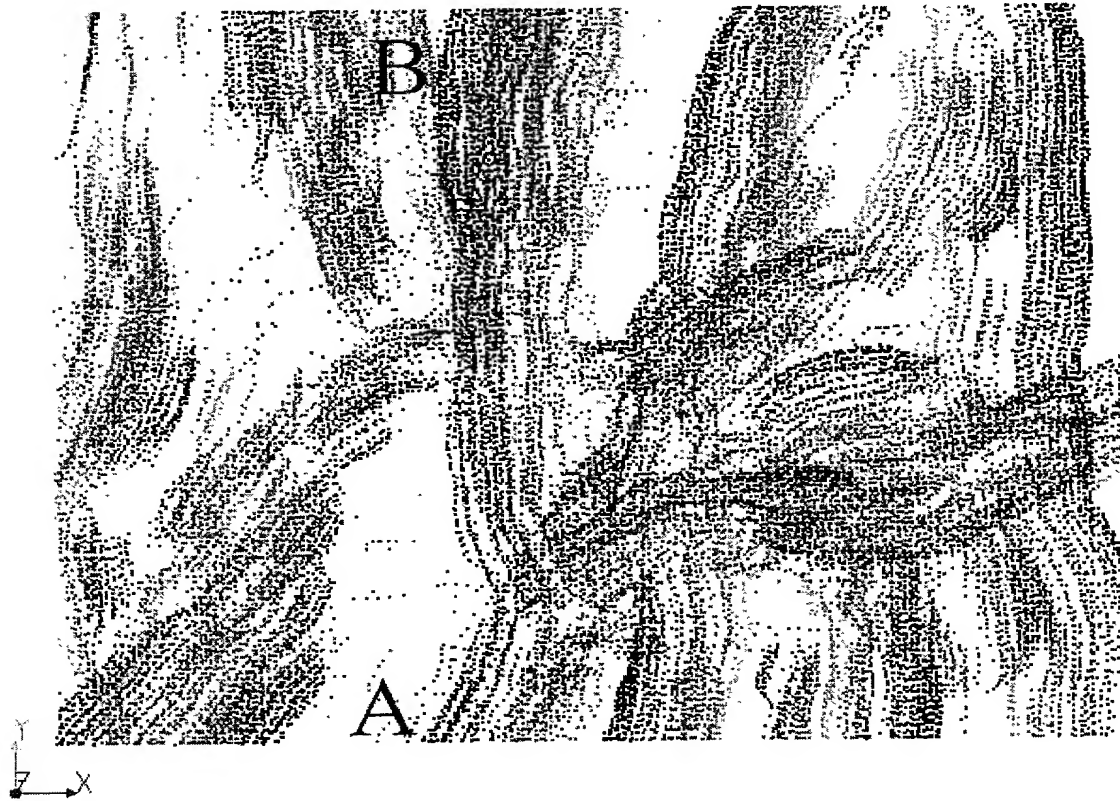


FIG. 20

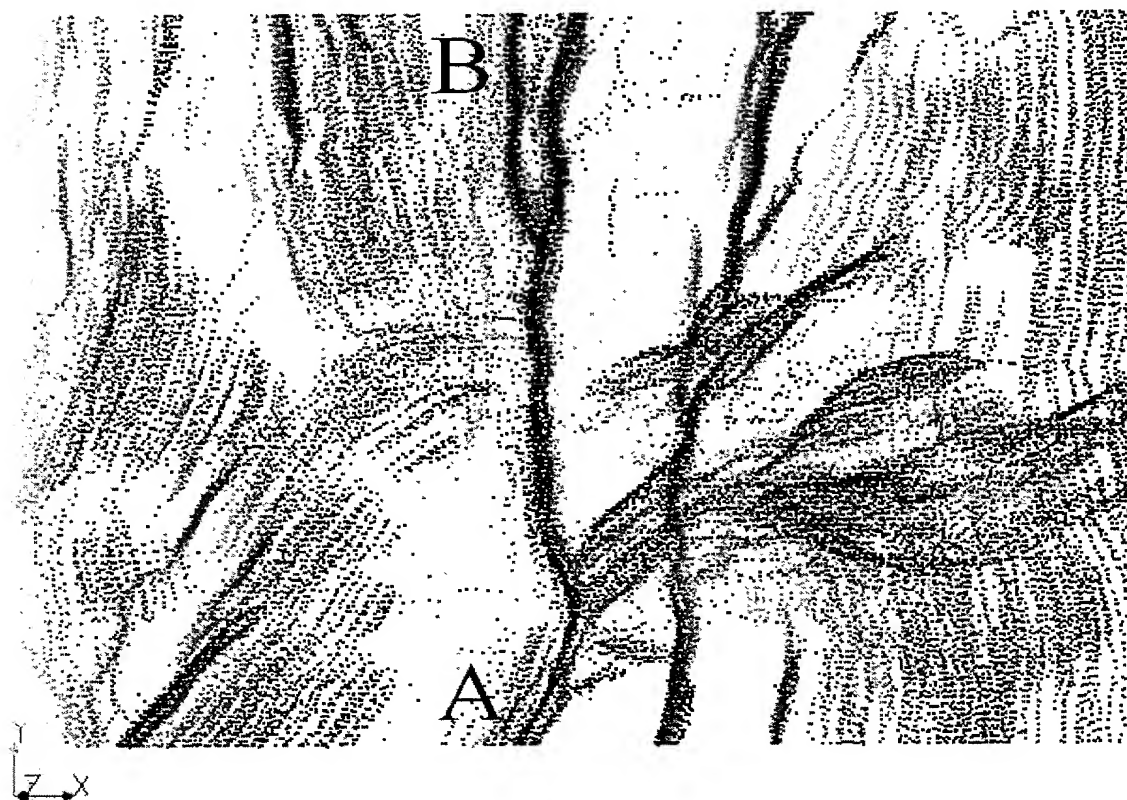


FIG. 21

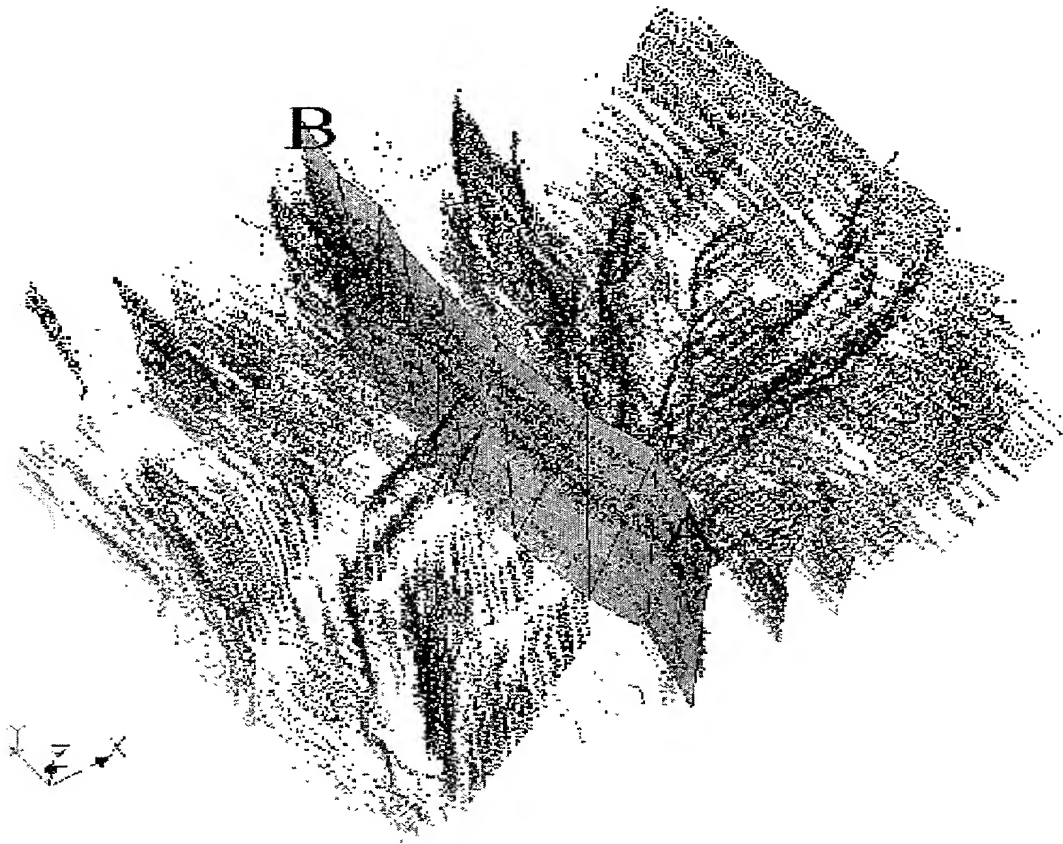


FIG. 22

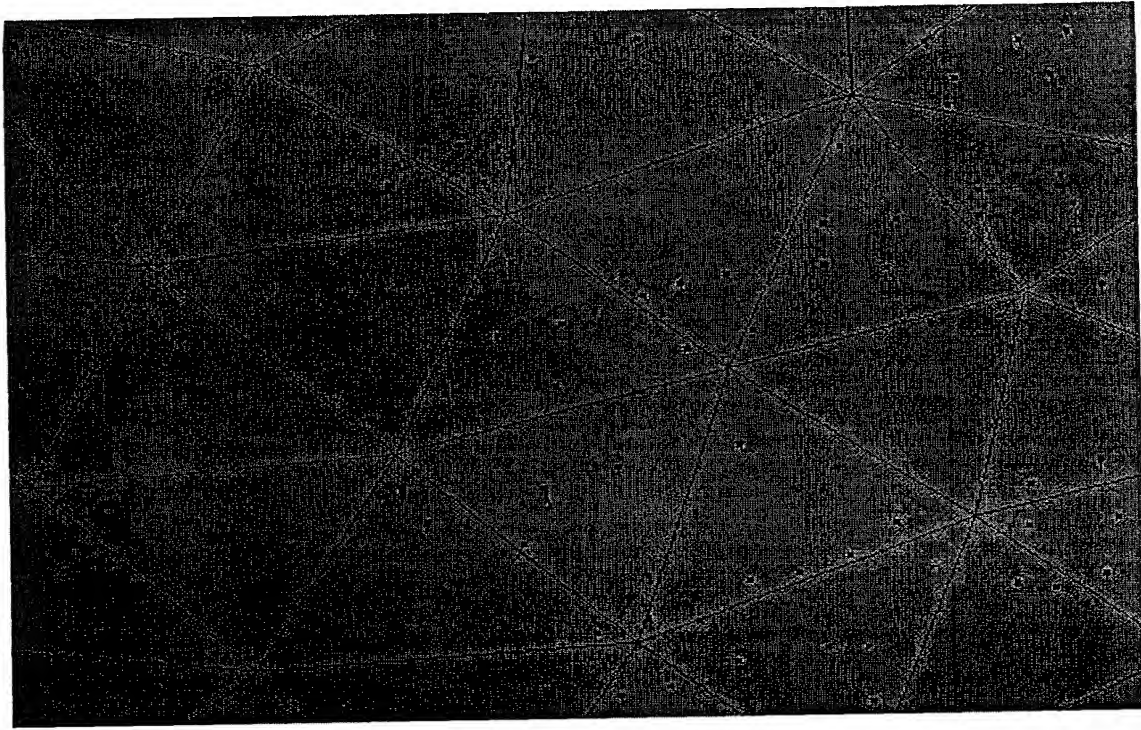


FIG. 23

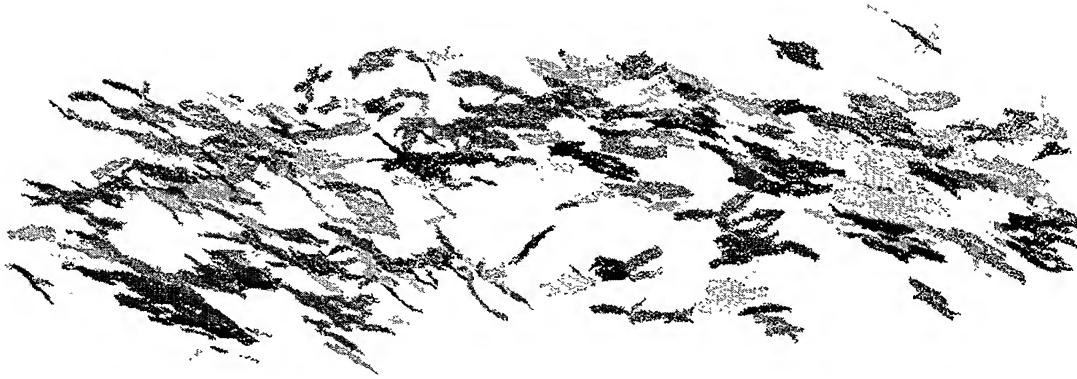


FIG. 24

FIG. 24



FIG. 25

FIG. 25

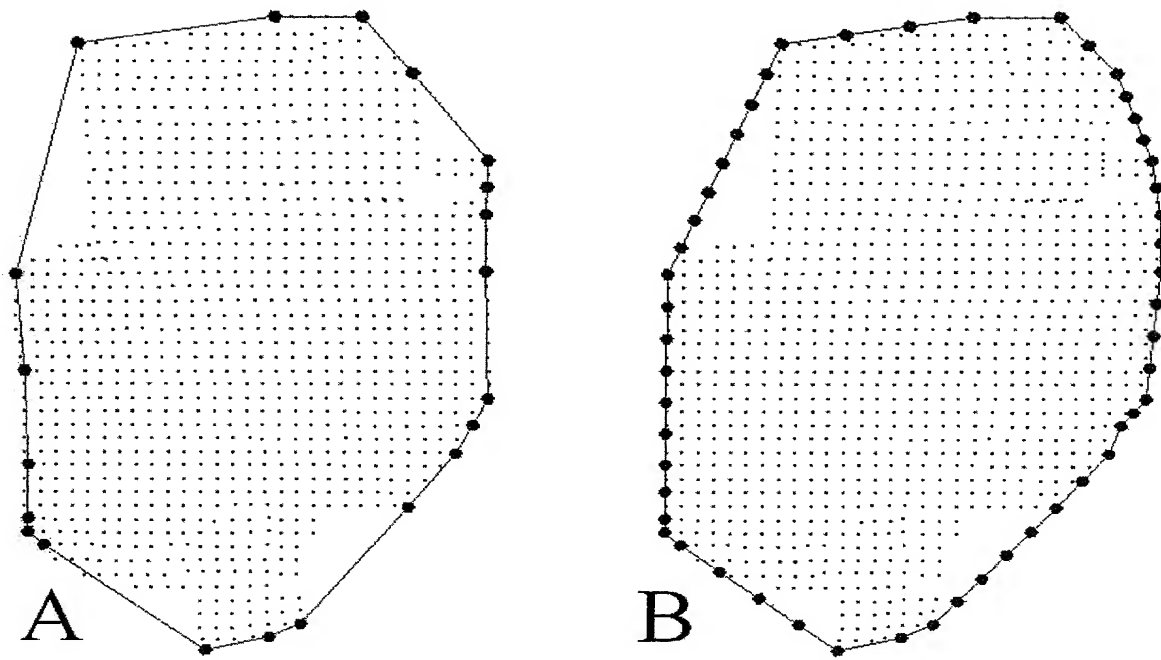


FIG. 26